RED HAT SUMMIT

10 YEARS *and counting*

SAN FRANCISCO | APRIL 14-17, 2014

Maximizing RAS with RHEL 7 beta DKU & Other Best Practices

Christoph Doerbeck
Principal Solutions Architect, Red Hat Inc.

Karl Abbott
Senior Technical Account Manager, Red Hat Inc.

Linda Wang
Senior Software Engineering Manager, Red Hat Inc

redhat.

10 YEARS *and counting*

SAN FRANCISCO | APRIL 14-17, 2014

Christoph Doerbeck covers “General Best Practices”

- **logs & monitoring**
- **more monitoring agents (smartd, mcelog, etc...)**
- **dm multipath & nic bonding**
- **cgroups & selinux**
- **sysrq trigger**

Karl Abbott covers Optimizing You Interactions with CEE

- **sosreports**
- **Kexec / Kdump & VMCore Analysis**
- **ABRT**
- **BOMGAR & redhat-support-tool**

Linda Wang covers Dynamic Kernel Update (DKU)

Avoiding Common Outages

Proactive – Before Something Fails

- **Monitor, Detect & Repair**
- **Resource Constraints**: cpu load, memory consumption, disk capacity, etc...
- **Recoverable HW failures**: cpu, memory, disk i/o, network, power, fans, etc...
  - Hardware with built in Redundancy, Error Correction, etc...
- **Standard Builds**: are the proper tools installed & configured correctly everywhere?
- **Automation**

Reactive – After Something Fails

- **Software Failures**: Out of Resources, Bugs
- **Non-Recoverable HW Failures**
- **Collect Evidence & Engage Support**: if you weren't proactive, chances are you're missing key evidence to help us identify root-cause

Logs with rsyslogd

Synopsis

- rsyslogd (syslog) is the system logging service which collects & writes log messages based on defined parameters (facility + level)
  - facility names: auth, authpriv (for security information of a sensitive nature), cron, daemon, ftp, kern, lpr, mail, news, syslog, user, uucp, and local0-7
  - level names: alert, crit, debug, emerg, errinfo, notice, warning
- Provides simple configuration & customization for services & applications
- Can be centralized

Enablement

- chkconfig rsyslog on; service rsyslogd start
- configuration: /etc/rsyslogd.conf & /etc/rsyslog.d/*.conf

Logs with rsyslogd

Example

- Use **logger** to properly log messages from CLI or shell scripts

Additional References

- Rotate the logs with **logrotate**
  - config: /etc/logrotate.conf & /etc/logrotate.d

mcelog, edac, hwpoison & ras-utils

Synopsis

- –mcelog – extracts Machine Check Events from kernel ring buffer and writes to a human readable file (/var/log/mcelog).
- –Newer AMD processors do not support mcelog daemon
  - •mcelog-1.0pre3_20110718-0.14.el6 (RHEL 6.3) properly reports error on newer AMD processors. See enablement below.
- –Intel Ivy Bridge & Haswell support in RHEL 6.5
- –hwpoison: gracefully survive certain memory failures

Enablement

- –Intel: chkconfig mcelog on ; service mcelog start
- –AMD: lsmod | grep edac_mce_amd

mcelog, edac, hwpoison & ras-utils

Example

- load kernel module with **modprobe mce-inject**
- simulate MCE with **mce-inject**
  - WARNING – simulating a panic event, will panic your host

Additional Resources

- LWN article on HWPoison: https://lwn.net/Articles/348886/
- mcelog can also keep stats or trigger shell scripts on specific events
- Install ras-utils rpm (from “RHEL Server Optional”) for development & testing
  - mce-inject, aer-inject
- http://www.mcelog.org

smartd

Synopsis

- smartd is a daemon that monitors the Self-Monitoring, Analysis and Reporting Technology (SMART) system built into many ATA-3 and later ATA, IDE and SCSI-3 hard drives
- polls devices every 30 minutes (configurable), logging SMART errors and changes of SMART Attributes via the SYSLOG interface.

Enablement

- yum install smartmonutils
- chkconfig smartd on; service smartd start
- configuration: /etc/smartd.conf

smartd

```
smartd[6157]: Device: /dev/sdf [SAT], opened
smartd[6157]: Device: /dev/sdf [SAT], ST2000DM001-1CH164, S/N:S1E0T9VM, WWN:5-0
smartd[6157]: Device: /dev/sdf [SAT], found in smartd database: Seagate Barracu
smartd[6157]: Device: /dev/sdf [SAT], is SMART capable. Adding to "monitor" lis
smartd[6157]: Monitoring 6 ATA and 0 SCSI devices
smartd[6157]: Device: /dev/sdf [SAT], 88 Currently unreadable (pending) sectors
smartd[6157]: Sending warning via mail to root ...
smartd[6157]: Warning via mail to root: successful

smartd[6169]: Device: /dev/sdf [SAT], 88 Currently unreadable (pending) sectors
smartd[6169]: Device: /dev/sdf [SAT], 88 Offline uncorrectable sectors
```

Examples

- View a summary of information:
  - smartctl -Ai /dev/sda
- View the error log:
  - smartctl -l error /dev/sda
- Start the SMART short & long test
  - smartctl -t short /dev/sda
  - smartctl -t long /dev/sda

Monitoring Logs

```
--------------------- Smartd Begin -----------------------
Currently unreadable (pending) sectors detected:
       /dev/sdf [SAT] - 9 Time(s)
       88 unreadable sectors detected
Offline uncorrectable sectors detected:
       /dev/sdf [SAT] - 9 Time(s)
       88 offline uncorrectable sectors detected
Warnings:
 Sending warning via mail to root ... - 2 Time(s)
       Warning via mail to root: successful - 2 Time(s)
```

Synopsis

- Get alerted & react when bad things happen
- Opensource Options: logwatch, Nagios, Zabbix, plenty more...
- Well established 3rd party tools: BMC Patrol, HP OpenView, IBM Tivoli, etc...

Additional References

- Don't forget to rotate additional log files with **logrotate**
  - config: /etc/logrotate.conf & /etc/logrotate.d

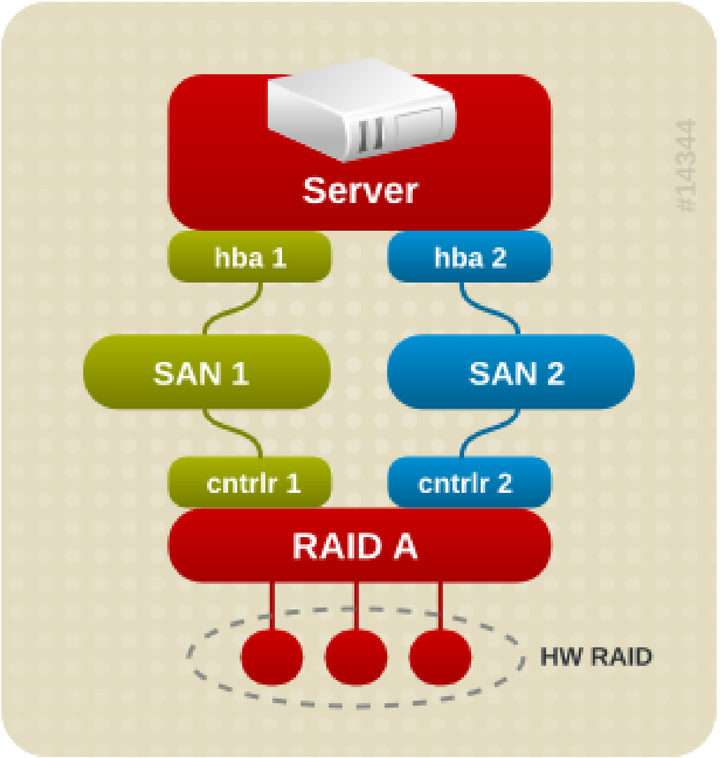
dm-multipath

Synopsis

- Storage I/O **redundancy** and/or **increased throughput**
- Discovers & configures multiple I/O paths between server & storage arrays
- “Paths” include separate cables, switches & controllers
- Creates a new device with the aggregated paths

Enablement

- yum install device-mapper-multipath
- mpathconf --enable --with_multipathd y
- service multipathd start
- Configuration File: /etc/multipath.conf



dm-multipath

Some Things to Know

- Modifying config after daemon is started requires '**service multipath reload**'
- Some Key Configuration Options
  - **blacklist** devices to exclude them from multipath detection
  - **find_multipaths** (RHEL 6) intelligent device discovery (/etc/multipath/wwids)
  - **user_friendly_names**
  - **path_selector :**
    - **round-robin**: loops thru every path in path group
    - **queue-length**: path with least number of outstanding I/O requests.
    - **service-time**: path with shortest service time
  - **path_grouping_policy & prio :** assigns priority to paths (ex: Clariion)

dm-multipath

Additional Resources

- Quick Guide:
  https://access.redhat.com/site/solutions/3689
- Comprehensive Guide:
  https://access.redhat.com/site/documentation/en-US/Red_Hat_Enterprise_Linux/6/html/DM_Multipath/index.html
- Configuration Details:
  https://access.redhat.com/site/documentation/en-US/Red_Hat_Enterprise_Linux/6/html/DM_Multipath/config_file_defaults.html#tb-config_defaults

channel (nic) bonding

Synopsis

- Combines two or more network interfaces to form a single "bonded" interface
- **Redundancy** and/or **Increased throughput**

Enablement

- Configure the bonded interface
- Configure network interfaces

../network-scripts/ifcfg-bond0

```
DEVICE=bond0
IPADDR=192.168.0.1
NETMASK=255.255.255.0
ONBOOT=yes
BOOTPROTO=none
USERCTL=no
BONDING_OPTS="bonding params"
NM_CONTROLLED=no
```

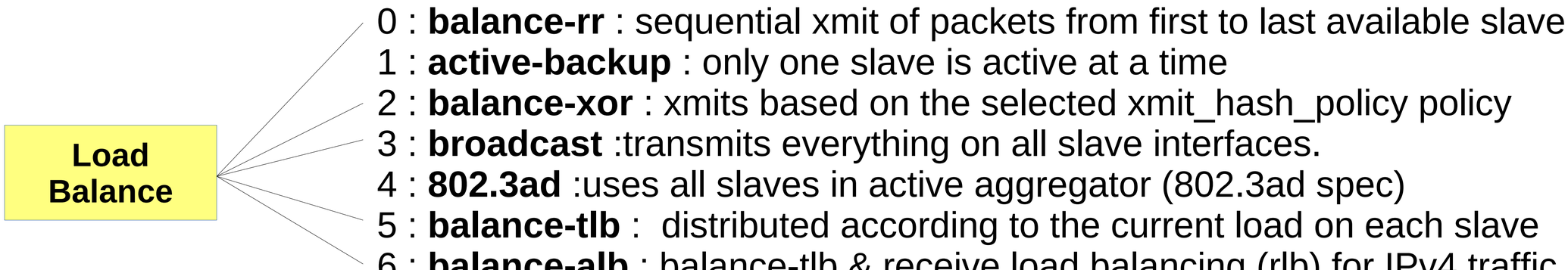
../network-scripts/ifcfg-ethN

```
DEVICE=ethN
BOOTPROTO=none
ONBOOT=yes
MASTER=bond0
SLAVE=yes
USERCTL=no
```

channel (nic) bonding

Example

- Modes (all provide fault tolerance):



Additional Resources

- Red Hat Enterprise Linux 6 Deployment Guide
  - https://access.redhat.com/site/documentation/en-US/Red_Hat_Enterprise_Linux/6/html/Deployment_Guide/s2-networkscripts-interfaces-chan.html
- How do I configure bonding device on Red Hat Enterprise Linux (RHEL)?
  - https://access.redhat.com/site/articles/172483#Bonding_modes_on_Red_Hat_Enterprise_Linux

CGroups

Synopsis

- Introduced in RHEL 6
- Dynamic allocation of resources
  - **processes, memory, storage & network**

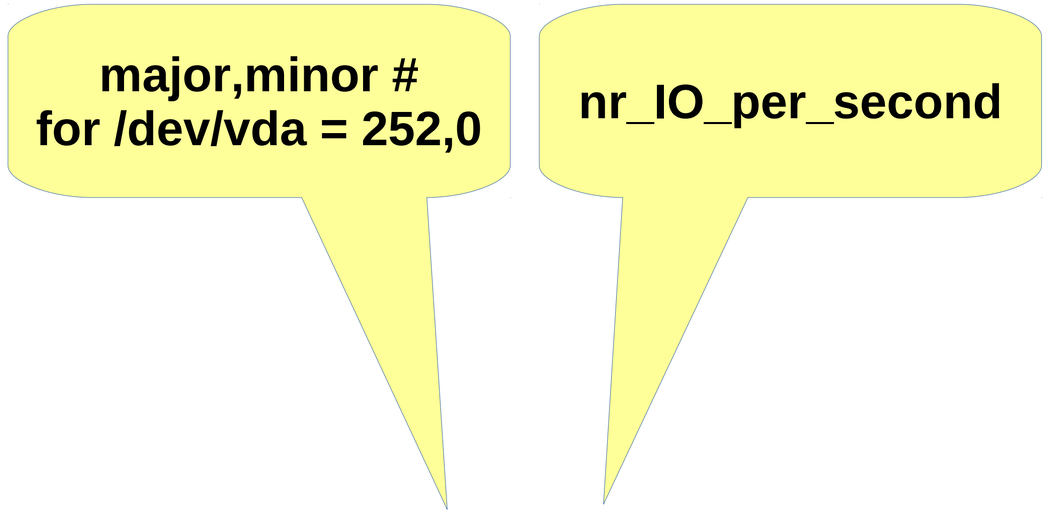
Enablement

- yum install libcgroup
- chkconfig cgconfig on
- service cgconfig start

10 subsystems that cgroups can leverage (RHEL 6.5)

blkio	: limits i/o access to & from block devices (ie: disks, ssd, USB, etc...)
cpu	: uses scheduler to provide cgroup access
cpuacct	: generate reports on CPU resrouces used by tasks
cpuset	: assigns individual CPUs & memory nodes
devices	: allows or denies access to devices
freezer	: suspends or resumes tasks
memory	: sets limits & reports on memory use by task
net_cls	: tags network packets within a classid (for use with tc)
net_prio	: set priority of network traffic per nic interface
ns	: namespace subsystem

CGroups



Example

- –create: **cgcreate -g blkio:/grpfoo**
- –config: **cgset -r blkio.throttle.read_iops_device="252:0 100" /grpfoo**
- –test: **cgexec -g blkio:grpfoo tar cf /dev/null --totals /usr**

Additional Resources

- –Red Hat Enterprise Linux 6.5 Resource Management Guide
  - https://access.redhat.com/site/documentation/en-US/Red_Hat_Enterprise_Linux/6/html-single/Resource_Management_Guide/

SELinux

Synopsis

- Mandatory Access Control (ACL) mechanism in the Linux kernel
- Allows operations after checking standard discretionary access controls
- Reduced vulnerability to privilege escalation attacks
- Decisions based on all available information, such as an SELinux user, role, type, and optionally a level

Enablement

- config: /etc/sysconfig/selinux
- modes: enforcing, permissive, disabled
- types: targeted, mls (multi-level-security)

SELinux

Example

- run **sestatus** to determine if SELinux is enabled
- run **ls -Z *filename*** to view SELinux context of a file / directory
- if enabled, **auditd** logs messages (denials) to /var/log/audit/audit.log

Additional Resources

- Security-Enhanced Linux User Guide
  - https://access.redhat.com/site/documentation/en-US/Red_Hat_Enterprise_Linux/6/html/Security-Enhanced_Linux/
- http://danwalsh.livejournal.com/20931.html
- Tools to diagnose SELinux problems: **setroubleshoot**
  - also logs to syslog (/var/log/messages)

SYSRQ Trigger

Synopsis

- –best (sometimes only) way to determine what a machine is really doing
- –sends signal requesting diagnostic information to kernel
- –system appears "hung" or diagnosing elusive, transient kernel-related problems

Enablement

- –/etc/sysctl.conf and modify “kernel.sysrq = 1”
- –`sysctl -w kernel.sysrq=1`
- –additional config for remote management cards (ex: ilo, drac, etc...)

SYSRQ Trigger

m	dump information about memory allocation
t	dump thread state information
p	dump current CPU registers and flags
c	intentionally crash the system (useful for forcing a disk or netdump)
s	immediately sync all mounted filesystems
u	immediately remount all filesystems read-only
b	immediately reboot the machine
o	immediately power off the machine (if configured and supported)
f	start the Out Of Memory Killer (OOM)
w	dumps tasks in uninterruptable (blocked) state [Introduced with kernel 2.6.32]

Example

- If system is reponsive
  - echo 'm' > /proc/sysrq-trigger
- If system is not responsive (appears hung)
  - on system console issue “SysRq m”
- Output is written to the kernel ring buffer & system console
- Normally logged via syslog to /var/log/messages.

Additional References

- https://access.redhat.com/site/articles/231663

10 YEARS *and counting*

SAN FRANCISCO | APRIL 14-17, 2014

SUPPORTING SUCCESS. EXCEEDING EXPECTATIONS.

Optimizing your interactions with CEE

redhat.


WE CAN DO MORE
WHEN WE WORK
TOGETHER

WHAT TO INSTALL BEFORE IT BREAKS

Software to have installed for a smoother support experience.

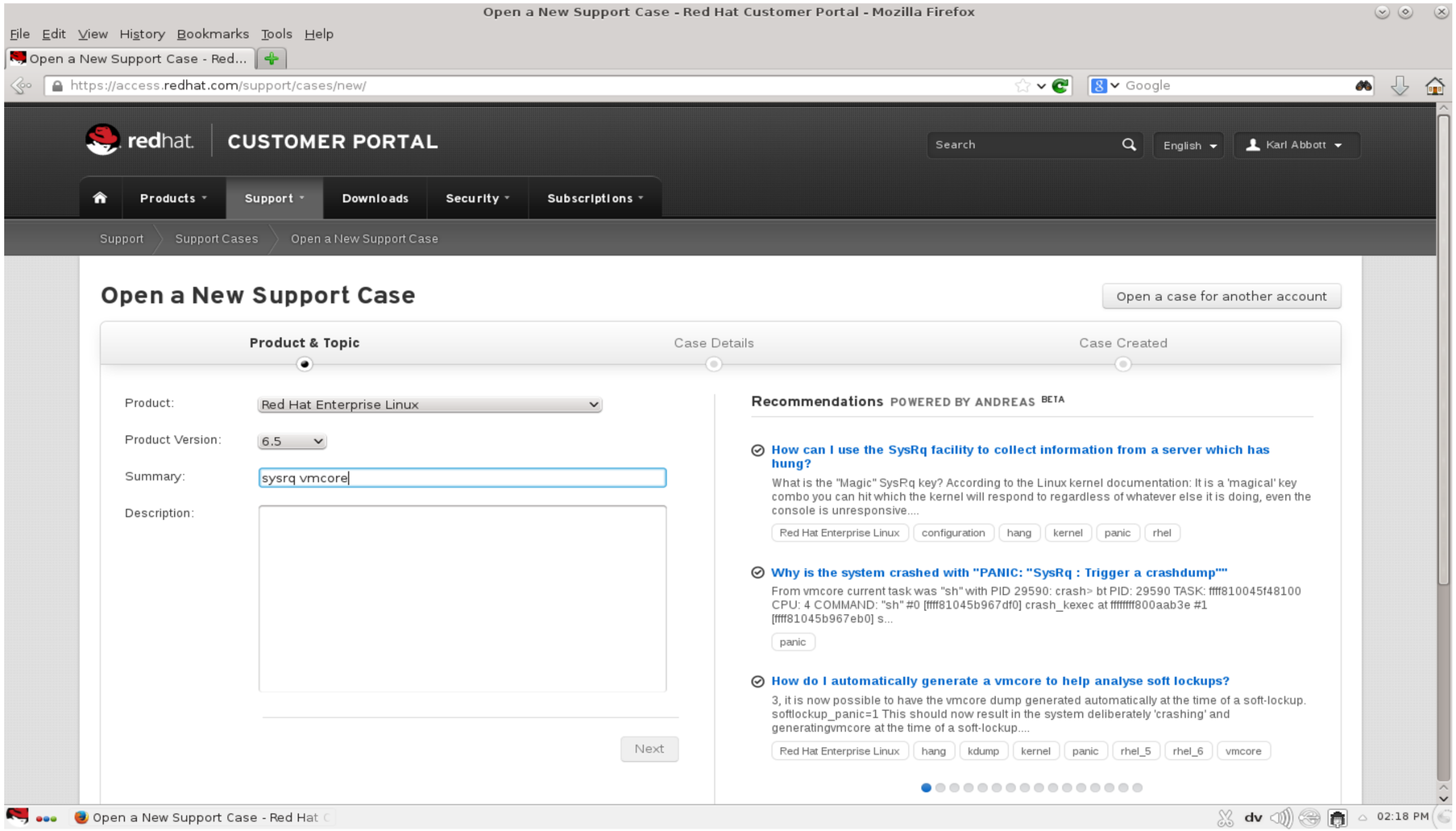
- sosreport
- kexec/kdump
- spacewalk-debug
- crash
- redhat-support-tool
- subscribe to the debuginfo channel!

RECOMMENDATIONS BY ANDREAS

Putting the Customer Portal to work for you!

- Open a new case and Andreas gets to work.

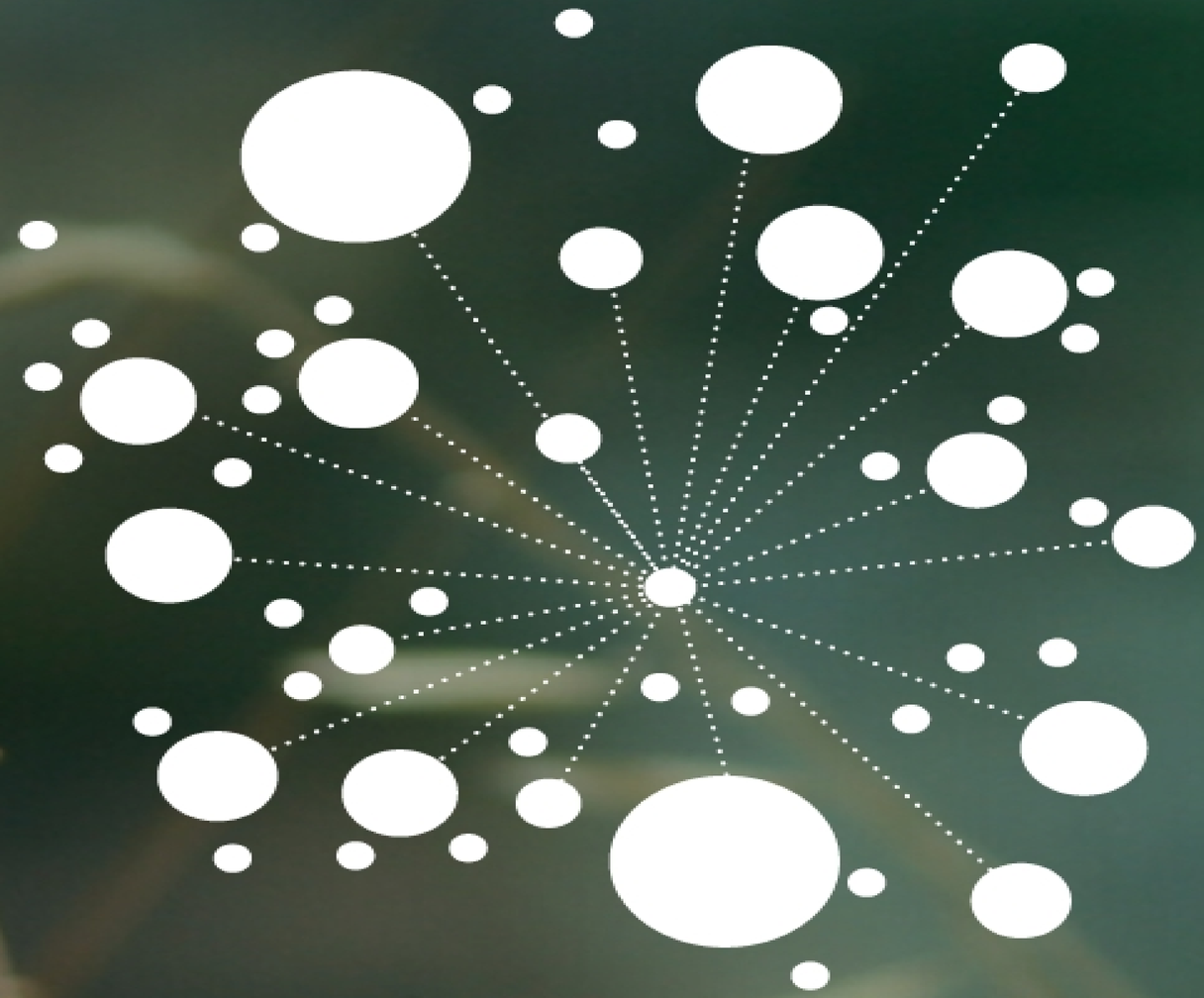
RECOMMENDATIONS BY ANDREAS



RECOMMENDATIONS BY ANDREAS

- See a suggestion that works for you? How did we know?
- KCS (Knowledge Centered Support) articles power Andreas.

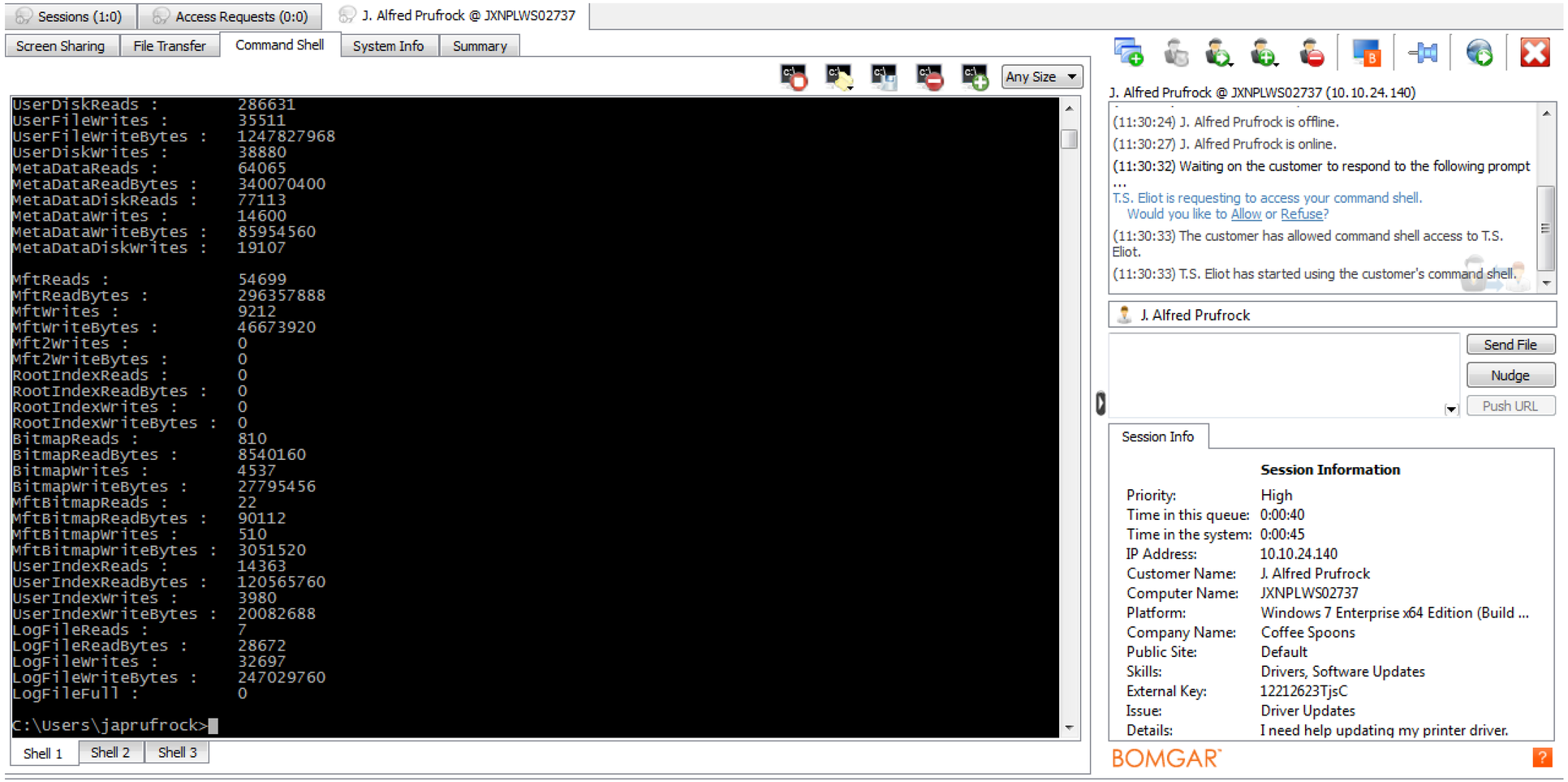
RED HAT KNOWLEDGE-CENTERED SUPPORT



REMOTE SUPPORT SESSIONS WITH BOMGAR

- Remote support capability.
- Red Hat can see your screen and work with you over the phone!
- For more information, see
  - https://access.redhat.com/site/solutions/412473
  - https://access.redhat.com/site/articles/255443

REMOTE SUPPORT SESSIONS WITH BOMGAR



PLEASE PROVIDE A SOSREPORT

Uses of sosreport

- Gather most commonly requested data points.
- Very important for understanding the context of an issue.
- For more information, see:
  - https://access.redhat.com/site/solutions/3592

SPACEWALK-DEBUG

Satellite's equivalent of a sosreport

- Spacewalk-debug provides Satellite specific information.
- For more information, see:
  - https://access.redhat.com/site/solutions/11047

ABRT

Detect and report problems as they happen.

- Automatic Bug Reporting Tool.
- Captures application crashes.
- Better integration with Satellite and Customer Portal in the future.
- For more info, see:
  - https://access.redhat.com/site/articles/642323
  - https://access.redhat.com/site/articles/718083

ABRT

Detect and report problems as they happen.

- Automatic Bug Reporting Tool.
- Captures application crashes.
- Better integration with Satellite and Customer Portal in the future.
- For more info, see:
  - https://access.redhat.com/site/articles/642323
  - https://access.redhat.com/site/articles/718083

KEXEC / KDUMP

RHEL 5, 6, and 7 use KDUMP to capture vmcores.

- Setting up kdump requires:
  - Grub parameter 'crashkernel'.
  - Configuration file '/etc/kdump.conf'.
  - Disk space to dump to.
  - Can compress with “-d 31” on the core_collector line of kdump.conf.
  - For more information: https://access.redhat.com/site/solutions/6038

VMCORE

A snapshot of memory at the time your box panicked!

- Gives us the details of what happened.
- Increases the chance we will get a root cause.

VMCORE

But my box has 4 TB of RAM!

- vmcore files are large. They can be up to the size of the RAM of the box that crashed.
- Upload via ftp or work with Support to ship a drive.

VMCORE

How to get answers fast!

- Find the RIP and search the Customer Portal with it.
- No matches? Provide that to Red Hat Support!

RAS – Reliability - Analysis

Dynamic kernel updates

- Analysis of the code changes - Building
  - Object level comparison of kernel objects (ELF relocatable files)
  - How:
    - Compiled using the -ffunction-sections and -fdata-sections GCC flags.
  - Advantages:
    - There is a one-to-one relationship between function/object symbols and the sections that contain their data. This allows precise cherry picking of the code and data segments that need to be included in the output object.
    - This also allows for a simple memory comparison (memcmp) of the section to determine if a particular function or object has changed.

RAS – Reliability - Analysis

Dynamic kernel updates

- Analysis of the code changes - Building
  - Advantages:
    - Second, it isolates each text/rela section pair that corresponds to a particular function from changes in other functions. If each function is not in its own section, a change to one function can cause the entire shared text section to shift, resulting in “changes” to the shared text and rela regions in other functions.
    - Using -ffunction-sections avoids this unpleasantness by starting each function at offset 0 in its own section

RAS – Reliability - Analysis

Dynamic kernel updates

– Analysis of the code changes – Object Comparison

- Per-object file comparison
- Two object files being compared: the “base” version and the “patched” version
- Each object file is opened and parse into structure represent elements: sections and symbols
- Then a correlation comparison between the structures: a comparison of section header and a memcmp of the section data
- This process produces a preliminary set of changed elements that need to be included in the output object

RAS – Reliability - Analysis

Dynamic kernel updates

- Analysis of the code changes – Reachability Test
  - Once all of the changed and dependency sections have been marked, a “reachability” test is performed.
  - To confirm that all changed sections are reachable from a changed function
    - i.e. Cases such as modifications to statically declared data structures are caught by this test.
  - If the reachability test passes, we are now ready to generate the output object.

RAS – Reliability - Conversion

Dynamic kernel updates

- Analysis of the code changes – kpatchTransformation
  - Once we generated the output objects, two additional sections need to be added
    - __kpatch_patches and .rela__kpatch_patches
    - In these text sections, after linking done by the kernel module loader, will contain one entry for each function that needs to be patched
    - Each entry contains the address of the base function in the running kernel and the address of the patched function in the hot-patch kernel module.
  - The static linking of non-exported symbols in the symbol table
    - If not in the symbol table of the output object, for each global entry that isn't exported by the kernel, the symbol is looked up in vmlinux and add in.

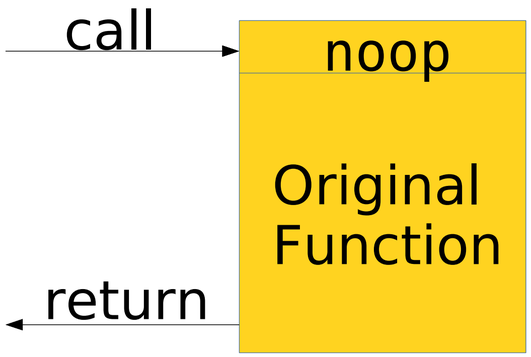
RAS – Reliability - Patching

Dynamic kernel updates

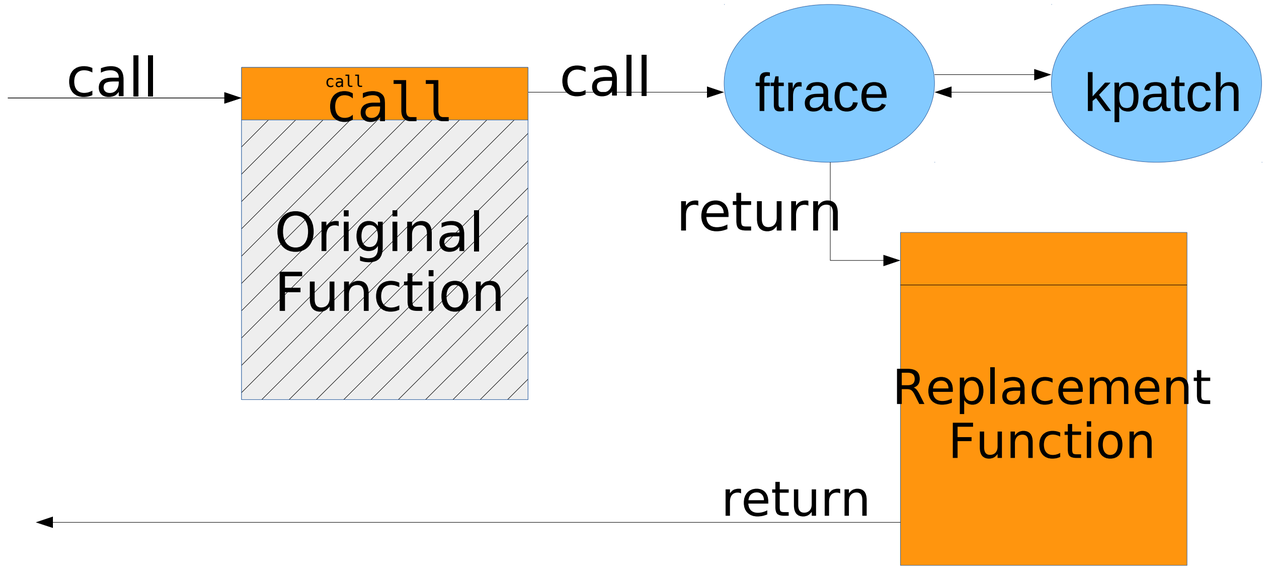
- Insertion of the changed object file
  - Quiscing the system to idle CPU's, verify activeness safety
  - Registered a trampoline function with ftrace
  - When ftrace hits the target function, trampoline function is called by ftrace immediately before the traget's original code is executed.
  - The the trampoline function then modifies the return instruction pointer (IP) address on the stack and return to ftrace, then restore the original function arguments and stack and continue on with the new function.

RAS - How it works:

Before patching:



After patching:



RAS - Servicability

Dynamic kernel updates

- Functional Support
  - Kexec Kdump/Crash will continue to work
  - A taint flag to identify the kenel that contains DKU modules
  - Tracepoint, perf, ftrace continue to work
  - Systemtap modules
  - Sosreport & ABRT will integrate
- System state will be preserved across reboot for presistancy